IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

Method and Apparatus for
Managing Data in a Gaming System

Inventors:

Jon Marcus Randall Whitten

Christopher M. Pirich

ATTORNEY'S DOCKET NO. MS1-768US

COPYRIGHT NOTICE

TECHNICAL FIELD

This invention relates to gaming systems, and more particularly, to the managing of data stored on a disk drive or other storage devices in a gaming system.

BACKGROUND

Gaming systems currently available on the market are capable of playing game discs, music CDs, and movie DVDs from a disc drive. For example, the Playstation® 2 gaming system from Sony Corporation provides the ability to play games, music, and video titles from a disc inserted in the console. These gaming systems have limited internal data storage capacity. Typically, the internal data storage is used to store system and configuration information, such as the local time, the language preference of the user, and other settings. Other data, such as saved game data and other game-specific data, is generally stored on a memory device that is external to the game console. For example, memory units that are inserted into a handheld game controller store game information for later retrieval by a game console. Existing gaming systems do not contain an internal non-removable hard disk drive for storing saved games and other information.

Microsoft Corporation recently announced its Xbox™ video gaming system that is equipped with a hard disk drive to enhance gaming, and broadband connectivity to facilitate online gaming. With these additions, significant amounts of data (e.g., saved game data from multiple game titles and multiple users of the gaming system) can be stored within the video gaming system using the hard disk drive. This new internal storage capability creates new issues with respect to storing and segregating different types of data on the hard disk drive. Further, data associated with a particular game title should be protected from inadvertent or intentional corruption by another game title or application.

Accordingly, there is a need for an improved data handling mechanism for gaming systems that include an internal data storage device, such as a hard disk drive.

SUMMARY

A gaming system includes a hard disk drive for storing various data. The gaming system applies a storage hierarchy to the hard disk drive to prevent unauthorized access to data stored on the hard disk drive. The hard disk drive is segregated into different logical regions such that each region stores a particular type of data and has specific data access policies. Each application that executes on the gaming system has a unique identifier and is provided with a separate area within certain regions for storing data related to the application. The separate areas provided for each application are based on the application's unique identifier. The gaming system also maintains a list of recently used nicknames to simplify user-entry of nicknames (e.g., a high-score entry for a game).

In the described implementation, the gaming system includes a game console and one or more controllers. The game console is equipped with a processor and a non-removable hard disk drive coupled to the processor. The game console may also include a memory, a portable media drive configured to communicate with a storage disc, one or more portable memory units, and broadband connectivity. In other implementations, the hard disk drive is configured to store game data, audio data, and video data.

A console application stored on the hard disk drive implements a user interface to allow user interaction with the gaming system. The game console boots into the console application upon power up or system reset. The console application also performs various functions necessary to operate the gaming system.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 illustrates an exemplary gaming system.

Fig. 2 is a block diagram of the gaming system.

Fig. 3 illustrates a network gaming system in which the Fig. 1 gaming system is connected via a network to other consoles and services.

Fig. 4 illustrates an example hard disk drive containing five separate regions.

Fig. 5 illustrates exemplary data stored in two different regions of a hard disk drive.

Fig. 6 illustrates exemplary saved game data stored in a user data region of a hard disk drive.

Fig. 7 is a flow diagram of a process that stores game-related data in appropriate regions of a hard disk drive.

Fig. 8 is a flow diagram of a process that manages various nicknames entered by users of the gaming system.

Fig. 9 illustrates a graphical user interface depicting an exemplary game selection menu.

Fig. 10 illustrates a graphical user interface depicting an example display of information related to saved games.

DETAILED DESCRIPTION

The method and apparatus described herein provides a hard disk storage hierarchy that is enforced to ensure proper storage of data and to prevent unauthorized access to stored data in a gaming system. The addition of a hard disk drive to a gaming system significantly increases the amount of data that can be stored within the gaming system. To manage this data, the hard disk drive's storage capacity is separated into different regions, where each region stores a particular type of data and has specific data access policies. The data access policies determine which applications are permitted to access a particular type of data. For example, a hard disk drive may contain a settings region, a user data region, an application data region, a utility region, and a console application region.

Each application that executes on the gaming system is provided with a separate area (such as a subdirectory) within the appropriate regions for storing data related to the application. Each application is restricted to viewing and accessing data in its own areas, thereby preventing accidental corruption of data

associated with different applications. A console application is executed by the gaming system in addition to other applications (such as games, music applications, and movie applications). The console application performs various functions necessary to operate the gaming system, such as implementing a user interface, performing configuration operations, and performing various management functions. Console application data and configuration settings are stored in separate regions of the hard disk drive that are not generally accessible to applications other than the user interface application and configuration applications. The gaming system also stores recently used nicknames for entry in, for example, high-score listings. A user of the gaming system can select among the stored nicknames rather than entering the same name each time a game is launched or each time a high score is achieved.

Fig. 1 shows an exemplary gaming system 100. The gaming system 100 includes a game console 102 and up to four controllers, as represented by controllers 104(1) and 104(2). The game console 102 is equipped with an internal hard disk drive and a portable media drive 106 that supports various forms of portable storage media as represented by optical storage disc 108. Examples of suitable portable storage media include DVD, CD-ROM, game discs, and so forth.

The game console 102 has four slots 110 on its front face to support up to four controllers, although the number and arrangement of slots may be modified. A power button 112 and an eject button 114 are also positioned on the front face of the game console 102. The power button 112 switches power to the game console and the eject button 114 alternately opens and closes a tray of the portable media drive 106 to allow insertion and extraction of the storage disc 108.

The game console 102 connects to a television or other display (not shown) via A/V interfacing cables 120. A power cable 122 provides power to the game console. The game console 102 may further be configured with broadband capabilities, as represented by the cable or modem connector 124 to facilitate access to a network, such as the Internet.

Each controller 104 is coupled to the game console 102 via a wire or wireless interface. In the illustrated implementation, the controllers are USB (Universal Serial Bus) compatible and are connected to the console 102 via serial cables 130. The controller 102 may be equipped with any of a wide variety of user interaction mechanisms. As illustrated in Fig. 1, each controller 104 is equipped with two thumbsticks 132(1) and 132(2), a D-pad 134, buttons 136, and two triggers 138. These mechanisms are merely representative, and other known gaming mechanisms may be substituted for or added to those shown in Fig. 1.

A memory unit (MU) 140 may be inserted into the controller 104 or the game console 102 to provide additional and portable storage. Portable memory units enable users to store game parameters and port them for play on other consoles. For example, a user can save a game to a memory unit 140 using a particular game console then use that saved game data with a game executed on a different game console. In the described implementation, each controller is configured to accommodate two memory units 140, although more or less than two units may be employed in other implementations. A particular game console 102 may be configured to accommodate any number of memory units 140.

The gaming system 100 is capable of playing, for example, games, music, and videos. With the different storage offerings, titles can be played from the hard disk drive or the portable medium 108 in drive 106, from an online source, or from

a memory unit 140. A sample of what the gaming system 100 is capable of playing back include:

1. Game titles played from CD and DVD discs, from the hard disk drive, or from an online source.

2. Digital music played from a CD in the portable media drive 106, from a file on the hard disk drive (e.g., Windows Media Audio (WMA) format), or from online streaming sources.

3. Digital audio/video played from a DVD disc in the portable media drive 106, from a file on the hard disk drive (e.g., Active Streaming Format), or from online streaming sources.

Fig. 2 shows functional components of the gaming system 100 in more detail. The game console 102 has a central processing unit (CPU) 200 and a memory controller 202 that facilitates processor access to various types of memory, including a flash ROM (Read Only Memory) 204, a RAM (Random Access Memory) 206, a hard disk drive 208, and the portable media drive 106. The CPU 200 is equipped with a level 1 cache 210 and a level 2 cache 212 to temporarily store data and hence reduce the number of memory access cycles, thereby improving processing speed and throughput.

The CPU 200, memory controller 202, and various memory devices are interconnected via one or more buses, including serial and parallel buses, a memory bus, a peripheral bus, and a processor or local bus using any of a variety of bus architectures. By way of example, such architectures can include an Industry Standard Architecture (ISA) bus, a Micro Channel Architecture (MCA)

bus, an Enhanced ISA (EISA) bus, a Video Electronics Standards Association (VESA) local bus, and a Peripheral Component Interconnects (PCI) bus, also known as a Mezzanine bus.

As one suitable implementation, the CPU 200, memory controller 202, ROM 204, and RAM 206 are integrated onto a common module 214. In this implementation, ROM 204 is configured as a flash ROM that is connected to the memory controller 202 via a PCI (Peripheral Component Interconnect) bus and a ROM bus (neither of which are shown). RAM 206 is configured as multiple DDR SDRAM (Double Data Rate Synchronous Dynamic RAM) that are independently controlled by the memory controller 202 via separate buses (not shown). The hard disk drive 208 and portable media drive 106 are connected to the memory controller via the PCI bus and an ATA (AT Attachment) bus 216.

A 3D graphics processing unit 220 and a video encoder 222 form a video processing pipeline for high speed and high resolution graphics processing. Data is carried from the graphics processing unit 220 to the video encoder 222 via a digital video bus (not shown). An audio processing unit 224 and an audio codec (coder/decoder) 226 form a corresponding audio processing pipeline with high fidelity and stereo processing. Audio data is carried between the audio processing unit 224 and the audio codec 226 via a communication link (not shown). The video and audio processing pipelines output data to an A/V (audio/video) port 228 for transmission to the television or other display. In the illustrated implementation, the video and audio processing components 220-228 are mounted on the module 214.

Also implemented on the module 214 are a USB host controller 230 and a network interface 232. The USB host controller 230 is coupled to the CPU 200

and the memory controller 202 via a bus (e.g., PCI bus) and serves as host for the peripheral controllers 104(1)-104(4). The network interface 232 provides access to a network (e.g., the Internet, home network, etc.) and may be any of a wide variety of various wired or wireless interface components including an Ethernet card, a modem, a Bluetooth module, a cable modem, and the like.

The game console 102 has two dual controller support subassemblies 240(1) and 240(2), with each subassembly supporting two game controllers 104(1)-104(4). A front panel I/O subassembly 242 supports the functionality of the power button 112 and the eject button 114, as well as any LEDs (light emitting diodes) or other indicators exposed on the outer surface of the game console. The subassemblies 240(1), 240(2), and 242 are coupled to the module 214 via one or more cable assemblies 244.

Eight memory units 140(1)-140(8) are illustrated as being connectable to the four controllers 104(1)-104(4), i.e., two memory units for each controller. Each memory unit 140 offers additional storage on which games, game parameters, and other data may be stored. When inserted into a controller, the memory unit 140 can be accessed by the memory controller 202. Additionally, one or more memory units 140 may be inserted into game console 102 and accessed by the memory controller 202.

A system power supply module 250 provides power to the components of the gaming system 100. A fan 252 cools the circuitry within the game console 102.

The game console 102 implements a uniform media portal model that provides a consistent user interface and navigation hierarchy to move users through various entertainment areas. The portal model offers a convenient way to

access content from multiple different media types—game data, audio data, and video data—regardless of the media type inserted into the portable media drive 106.

To implement the uniform media portal model, a console user interface (UI) application 260 is stored on the hard disk drive 208. When the game console is powered on, various portions of the console application 260 are loaded into RAM 206 and/or caches 210, 212 and executed on the CPU 200. The console application 260 presents a graphical user interface that provides a consistent user experience when navigating to different media types available on the game console. Thus, the hard disk drive 208 (and the data stored thereon) is an important part of the initialization process. If the hard disk drive 208 is not functioning properly, the gaming system 100 may not boot successfully.

The gaming system 100 may be operated as a standalone system by simply connecting the system to a television or other display. In this standalone mode, the gaming system 100 allows one or more players to play games, watch movies, or listen to music. However, with the integration of broadband connectivity made available through the network interface 232, the gaming system 100 may further be operated as a participant in a larger network gaming community.

Fig. 3 shows an exemplary network gaming environment 300 that interconnects multiple gaming systems 100(1), ..., 100(g) via a network 302. The network 302 represents any of a wide variety of data communications networks. It may include public portions (e.g., the Internet) as well as private portions (e.g., a residential Local Area Network (LAN)), as well as combinations of public and private portions. Network 302 may be implemented using any one or more of a wide variety of conventional communications media including both wired and

wireless media. Any of a wide variety of communications protocols can be used to communicate data via network 302, including both public and proprietary protocols. Examples of such protocols include TCP/IP, IPX/SPX, NetBEUI, etc.

In addition to gaming systems 100, one or more online services 304(1), ..., 304(s) may be accessible via the network 302 to provide various services for the participants, such as hosting online games, serving downloadable music or video files, hosting gaming competitions, serving streaming audio/video files, and the like. The network gaming environment 300 may further involve a key distribution center 306 that plays a role in authenticating individual players and/or gaming systems 100 to one another as well as online services 304. The distribution center 306 distributes keys and service tickets to valid participants that may then be used to form games amongst multiple players or to purchase services from the online services 304.

The network gaming environment 300 introduces another memory source available to individual gaming systems 100 – online storage. In addition to the portable storage medium 108, the hard disk drive 208, and the memory unit(s) 140, the gaming system 100(1) can also access data files available at remote storage locations via the network 302, as exemplified by remote storage 308 at online service 304(s).

Fig. 4 illustrates an exemplary hard disk drive 400 containing five separate regions. In this example, the five hard disk drive regions are: a settings region 402, a user data region 404, an application data region 406, a utility region 408, and a console application region 410. The regions shown in Fig. 4 are provided as an example. In alternate implementations, a hard disk drive may contain any number of regions.

As discussed below, each region stores a particular type of data and has specific data access policies that restrict data access to those applications that are intended to access that type of data. The five regions represent a logical segregation of the hard disk drive and do not necessarily correspond to physical divisions or physical partitions of the hard disk drive. One or more applications (e.g., the console application) executed by the gaming system maintain and support this logical segregation of the hard disk drive into multiple regions.

The settings region 402 is used to store system state and configuration information used by the gaming system. Third-party applications (such as game applications, music applications and movie applications) are denied direct access to the settings region 402. Applications requiring data stored in the settings region 402 request such data through API (application program interface) calls. The storage space allocated to the settings region 402 is managed by the console application 260. The settings region data is stored outside the file system and is accessed using sector level input and output commands. This configuration reduces the risk of data corruption and undesired access to settings data. The settings region 402 is protected using a checksum to ensure that the settings data is not corrupted and, if necessary, can survive a complete system restoration procedure.

The user data region 404 is used to store user data on the gaming system. User data may include, for example, game data saved by a user of the gaming system or picture files saved by a user. The data in the user data region 404 is organized into a specific hierarchy. Applications are required to conform to this hierarchy. In one implementation, applications must conform to the specific hierarchy before the application is certified to execute on the gaming system. If

the application is not certified to execute on the gaming system, the gaming system will reject any attempt to execute the application on the gaming system. Typically, the certification process is performed by the manufacturer of the gaming system.

In addition to storing user data in the user data region 404 of the hard disk drive 208, user data may also be stored in one or more memory units 140 (Fig. 1) coupled to one or more controllers 104 or the game console 102.

The application data region 406 is used to store persistent data used by various applications that are executed by the gaming system. The data stored in the application data region 406 is created at various times during the execution of the application. This data is typically stored without the user's knowledge. The data stored in region 406 allows the application to maintain data across multiple game sessions without requiring the data to be associated with a particular saved game. In a particular implementation, music files (e.g., WMA files) are saved in the application data region.

Exemplary data stored in the application data region 406 includes updated player rosters, game updates, new game levels, skid marks on a road, damage to a building or object, or other changes that are maintained by the application. Skid marks and damage to a building are examples of "environmental changes" to the game that are applied each time the game is executed, including executing saved games. Each application is provided with a separate storage area (e.g., subdirectory) within the application data region 406 to store its application data. Generally, only the application that saved the specific data in the application data region 406 is permitted to delete that specific data. However, if a user of the gaming system indicates a desire to delete all data related to a particular

application, the data in the application data region 406 associated with the particular application will be deleted.

In one implementation, the user data 404 and the application data 406 is stored in a single partition of the hard disk drive 400. This allows each application to use any portion of the data storage capacity provided by the partition. Each application uses a portion of the overall storage capacity of the partition. In this implementation, applications are not restricted to using a particular portion of the partition's total storage capacity. This configuration provides flexibility to the application (and the application developer) by allowing the application to use the amount of storage space desired. This configuration also reduces problems caused by limiting storage space allocated to each application. When applications are limited to a particular storage area, they may run out of storage space even though other applications are not using their entire storage allotment. Providing a single partition shared by all applications, reduces the likelihood that a particular application will not have sufficient storage space to execute properly.

The utility region 408 is used to store any data desired by the application. The utility region 408 can be used in any way by the application and the gaming system does not impose any restrictions on the use of the utility region by the application. Thus, each application is able to use its assigned utility region 408 in any manner that the application developer desires. Applications may use utility region 408 for caching data or creating a virtual memory space. The utility region 408 is for temporary storage of data. A particular application cannot be certain that the same data will be available the next time the application is executed.

In one implementation, three separate utility regions 408 are provided on the hard disk drive. Each utility region 408 contains 750 Megabytes of storage

space and is used by a different application to store various data. When an application is launched, the gaming system determines whether one of the three utility regions 408 contains information stored by the same application during a previous execution of the application. If so, that same utility region 408 and the data previously stored by the application is assigned to the application for use during the current execution of the application. If none of the utility regions 408 contain information stored by the same application, the system determines whether one of the utility regions is empty. If so, the empty utility region is assigned to the application. Otherwise, the utility region with the oldest data is cleared (i.e., all data in the utility region is deleted) and assigned to the application for use during the current execution of the application.

When an application begins using one of the three utility regions 408, a timestamp is applied to the particular utility region being used. The timestamp identifies the application using the utility region 408 (e.g., the application title) and the time that the application accessed the utility region. The three utility regions 408 are aged out using a least recently used (LRU) algorithm. If an application is requesting a utility region 408, but all utility regions contain data, none of which is associated with the requesting application, then the LRU algorithm deletes the data from the utility region having the oldest timestamp. This automated process relieves the user (and the application developer) from managing this temporary storage space.

An application may identify whether it wants the data in the utility region 408 saved for future reference or deleted when the application is terminated. If the data is deleted, the utility region will be made available to another application. If the data is saved, the data will be available to the application the next time it is

executed unless the LRU algorithm ages out the data stored in that utility region before the next execution of the application.

The console application region 410 is used to store various data used during execution of the console application, such as user interface data. Other applications are prevented from accessing the data stored in the console application region 410. In a particular implementation, the console application region 410 is stored in a separate partition of the hard disk drive 400 to reduce the likelihood that data stored in the console application region would become corrupted.

As shown in Fig. 4, various types of data are stored in the different regions of hard disk drive 400. However, other storage devices in the gaming system 100 may be restricted in the content they store. In one implementation, memory unit 140 is limited to storing saved game data associated with one or more games. Since the various gaming system settings and configuration data is specific to each gaming system, that data need not be distributed to other gaming systems (e.g., through portable memory unit 140). Instead, other gaming systems should rely on their own system settings and configuration data stored on their own hard disk drive or other internal storage device. Thus, when a user requests to save a game or other application data to memory unit 140, the data stored on the memory unit is limited to the data necessary to recreate the state of the game or other application at a later time. By limiting memory unit 140 to saving, for example, saved game data but no configuration data, the user is presented with a simple list of saved game data rather than a mixed list of saved game data and other configuration data that is not required by the user.

Fig. 5 illustrates exemplary data stored in two different regions, user data region 404 and application data region 406, of hard disk drive 400. Each region of the disk drive may be further segregated into different sections or storage areas (e.g., subdirectories) that are associated with a particular application. This further segregation ensures that data associated with a particular application is not accessed or modified by another application, thereby maintaining the integrity of the data associated with each application that is executed by the gaming system.

In the example of Fig. 5, the user data region 404 and the application data region 406 store data related to multiple game applications (referred to as "Game A", "Game B", etc.). User data region 404 includes four sections, one for each game's saved data, labeled Saved Game A Data 502, Saved Game B Data 504, Saved Game C Data 506, and Saved Game N Data 508. A particular user data region 404 may contain any number of sections, each of which is associated with a particular game or other application. A particular application is permitted to access data associated with that application, but prevented from accessing or modifying data that is associated with a different application. For example, Game A can access data stored in section 502. However, Game A is prevented from accessing data stored in any of sections 504, 506, or 508 because that data is associated with a different game.

Application data region 406 includes four sections, one for each application's saved data, labeled Game A Data 510, Game B Data 512, Game C Data 514, and Game N Data 516. As discussed above, the application data region is used by applications to store persistent data generated by the application. A particular application data region 406 can contain any number of sections, each of which is associated with a particular game or other application. Each application

is limited to accessing data from the section of region 406 that is associated with the application. For example, Game B can access Game B Data stored in section 512, but is prevented from accessing data in any of sections 510, 514, or 516 because that data is associated with a different game.

Fig. 6 illustrates exemplary saved game data 502 stored in the user data region 404 of hard disk drive 400. In this example, the saved game data 502 includes three sets of saved game data, labeled First Saved Game 602, Second Saved Game 604, and Third Saved Game 606. All three sets of saved game data 602-606 are related to the same game application (i.e., Game A). Each set of saved game data corresponds to a different "save game" command executed by a user of the gaming console. For example, each set of saved game data 602-606 may have been created on different days or created when the user of the gaming system reached a new level in the game. The saved game data 602-606 allows the user to restart the game application at the same location in the game (i.e., same game level, score, settings, etc.) as when the game was saved. A particular game application may create any number of sets of saved game data based on user requests to save the current state of the game. All saved game data related to a particular game (e.g., Game A) is stored in the same section (e.g., Saved Game A Data 502), which simplifies identification of the desired saved game.

Listed below are example directories used to store data on hard disk drive 400 in the gaming system. Similar directory structures can be used to store data on memory units 140 or other storage devices used by the gaming system.

ROOT\Udata\0FFFAB12 \FFE62\ <saved game files>
ROOT\Udata\0FFFAB12 \F4B1A\ <saved game files>

ROOT\Tdata \0FFFAB12 \ <saved data >

The first two directories identified above store various saved game data files for a particular game application. The "Udata" directory off from the root represents the user data region 404 of the hard disk drive 400. The next entry "OFFFAB12" is a subdirectory of the user data region 404, and is associated with a particular application. In this example, "OFFFAB12" represents the hash of the title of a game. Each game should generate a different hash value such that each subdirectory is unique. Alternatively, a unique game identifier may be used as the subdirectory name.

The next entries "FFE62" and "F4B1A" each represent a particular saved game subdirectory. Various data files associated with each saved game are stored in each subdirectory. A new subdirectory is created each time the user of the gaming system initiates a save game operation. The new subdirectory stores the various saved game data generated by that save game operation. The entries "FFE62" and "F4B1A" are generated by hashing various information about the saved game (such as the name of the saved game, the date the game was saved, the time the game was saved, or the game level when the game was saved). In one implementation, the entries "FFE62" and "F4B1A" are generated by hashing the name of the saved game. Alternatively a combined date and time code may be used to generate unique subdirectory names.

The third directory identified above stores additional data related to the game "0FFFAB12". The "Tdata" directory off from the root represents the application data region 406 of the hard disk drive 400. Since the data stored in the

application data region 406 generally applies to the game itself and not to particular saved games, there are no subdirectories off from "0FFFAB12".

A new subdirectory is created under "Udata" and "Tdata" each time a new application is executed by the gaming system. When an application is executed, the gaming system first checks for existing subdirectories associated with the application. If subdirectories already exist for the application, those subdirectories, and their corresponding data, are assigned to the application. However, if no subdirectories exist for the application, the console application creates the appropriate subdirectories for the application to use.

Although many different subdirectories may extend from "Udata" and "Tdata", a particular application only sees data and subdirectories contained under the application's particular subdirectory. Thus, the application is unable to see the entire directory structure or identify other applications that have associated subdirectories stored on the hard disk drive. These restrictions on data access prevent an application from intentionally or inadvertently corrupting data associated with another application.

In certain situations, the gaming system may allow an application to access data associated with a different application. For example, if a new version of an application is released, it may generate a different identifier than the previous version of the application, and thus be associated with a different set of subdirectories. In this example, the gaming system (e.g., the console application) can map or redirect the new version of the application to the subdirectories associated with the old version of the application, thereby allowing the new version of the application to access data stored by the old version of the application.

In another example, an application from a particular manufacturer may be permitted to access data associated with a different application from the same manufacturer. In this example, the manufacturer is given the responsibility of properly handling the data generated by its applications.

Fig. 7 is a flow diagram of a process 700 that stores game-related data in appropriate regions of a hard disk drive. The process 700 is implemented in software as computer-executable instructions that are executed by the CPU 200 to perform the operations illustrated as blocks. Initially, a game is launched from the gaming system (block 702). A game can be launched by inserting a game disc into portable media drive 106, selecting a game (or a saved game) from the hard disk drive 108 (via a user interface that identifies saved games), selecting a saved game from a memory unit 140, or selecting a game from an online source.

Next, the process 700 identifies a Game ID associated with the launched game (block 704). The Game ID is used by the gaming system to distinguish one game from another and ensure that different games access the appropriate sets of data from the hard disk drive 208 and other storage devices. In one implementation, the Game ID is assigned by the manufacturer of the gaming system to ensure that all Game IDs are unique. In another implementation, the Game ID is generated by creating a hash of the game title.

At block 706, the process 700 creates pointers or other mechanisms to identify the appropriate regions of the hard disk drive based on the Game ID. For example, a particular game may be permitted to access a particular portion of the user data region and a particular portion of the application data region. The pointers direct the application to the appropriate portions of the user data region and the application data region, but do not allow access to portions of the user data

region and the application data region that are associated with different games. A particular pointer may identify particular subdirectories on the hard disk drive, as discussed above.

The process 700 then determines whether a user of the gaming system has requested that the current game be saved on the hard disk drive (block 708). If the user has not requested to save the current game, execution of the game continues (block 710). If the user has requested that the current game be saved, the process saves the current state of the game as a set of data in the user data region of the hard disk drive (block 712). Additionally, other game-related data may be stored to the application data region of the hard disk drive (block 714). Although the saving of other game-related data is shown at block 714, such activity can occur at any point during the execution of the game. For example, if a car hits the wall of a race track during a race, the game may record the skid marks and damage to the wall in the application data region of the hard disk drive. Thus, the same skid marks and damage to the wall will be shown in the future when the same game is executed on the gaming system.

Fig. 8 is a flow diagram of a process 800 that manages various nicknames entered by users of the gaming system. The process 800 is implemented in software as computer-executable instructions that are executed by the CPU 200 to perform the operations illustrated as blocks. Nicknames are often used to assign names to characters in a game or to identify a user's high score. Initially, a game is launched from the gaming system (block 802). The process 800 retrieves a list of recently used nicknames (block 804). The list may be ordered such that nicknames recently used with the current game are displayed first, and nicknames used recently with other games are displayed later in the list. Next, the process

displays the list of recently used nicknames to the user of the gaming system (block 806). The user of the gaming system is then presented with the opportunity to select a nickname from the displayed list or create a new nickname for use in the current game. The process 800 determines whether the user selects a name from the nickname list or selects to enter a new nickname (block 808). If the user selected a name from the nickname list, the process branches to block 810, where the game is executed using the selected nickname. If the user selected entering a new nickname, the process continues to block 812, where the procedure displays a screen for entering a new nickname. After the user enters a new nickname, the process 800 executes the game using the new nickname and adds the new nickname to the list of recently used nicknames (block 814). If a high-score is achieved (or any other action requiring entry of the player's nickname), the selected nickname is automatically entered as the default name. The user is able to change the default name if they choose.

Fig. 9 illustrates a graphical user interface 900 depicting an exemplary game selection menu. The graphical user interface 900 is generated by the console application 260 executed by the CPU 200. The game selection menu is the area where the user can select from available game applications they have previously played on their gaming system. Although this example lists available games, similar user interfaces identify other types of applications, such as an audio player or a video player. The user interface 900 includes a list 902 of the games available on the gaming system. A game is an application that has been purchased, borrowed, or rented by the user and played on their gaming system at least one time. In Fig. 9, the games are shown in horizontal tiles or panes. It is noted that

other graphical themes may be alternatively used to represent available games, such as a bookshelf, a toy box, or the like.

The user interface 900 also includes an orb 904 depicting an image of the currently selected game title and a text panel 906 with information about the selected game. In the illustrated example, the game "Starcraft" is highlighted, resulting in an image of a character from the game "Starcraft" being depicted in orb 904 and information pertaining to this game being presented in text panel 906. The game developer is given control of the contents of the orb 904 and text panel 906, so the information will vary from one game to another.

A piece of descriptive text 908 (i.e., "n games") is positioned beside the main legend "Games" to indicate the number of games in the list. The list 902 displays a limited number of games (e.g., eight titles). When a user first enters the games collection after purchasing their gaming system, there will be zero titles in the list 902. To represent this, the descriptive text 908 states "0 games" and the text panel 906 offers a short statement telling the user that future games played on the console will appear in this area. As the user plays games, they are added to the list 902. When the descriptive text 908 indicates that there are more games than shown on list 902 (e.g., $n > 8$), up/down scroll arrows are added to the list 902 to indicate that there are additional titles not currently shown on the list.

As noted above, the game developer provides the data used to fill the orb 904 and text panel 906. When the user plays a game on the gaming system for the first time, a number of data elements are copied into the application data region of the hard disk drive 208 for use by the application during future executions of the application.

The user can move among games in list 902 by using the up and down directions of the thumbstick, or some other pre-defined control mechanism. The list 902 may be configured to wrap or not wrap when the user reaches the top or the bottom of the list. A select element 910 allows the user to select the highlighted game from list 902 using the "A" button on the controller. A back element 912 facilitates navigation back to the previous menu in the user interface. The back element 912 is chosen by pressing the "B" button on the controller, as visually aided by the letter "b" in the element 912.

Fig. 10 illustrates a graphical user interface 1000 depicting an example display of information related to saved games. The user interface 1000 provides a view of all content data that is currently available on the selected memory device, such as the hard disk drive. The user interface 1000 depicts a flat list 1002 of games and their corresponding saved games, soundtracks and their associated tracks, and video clips that are stored on the selected memory device. Each file is represented by small orbs 1004 arranged in horizontal panes. Each orb has an image that identifies the contents, such as a game image or the last scene before the game was saved. The file 1004 has an associated number that denotes the total size of the saved game in blocks.

In the games context, the list of files 1002 is formatted such that the game graphic is situated in an orb 1006 located near the title of the game title (e.g., "Starcraft"). The orb 1006 is selectable and upon selection, performs a multi-select on all of the saved games for the selected game. Each saved game is selectable as well by navigating to the desired orb 1004. As before, navigation can be achieved by using the left, right, up, and down directions of the thumbstick, or

other mechanism. In one implementation, the saved game orbs 1004 are sorted by most recently saved within each game entry.

A text panel 1008 offers a richer description of the saved game, audio track, or video clip that is currently focused. In the game context, this description might include the following information:

- Saved Game
    - 2D image associated with the saved game
    - Game that the saved game belongs to
    - Saved game name
    - The location (e.g., level) in the game or current mission
    - Date and Time the game was saved
    - Total size of the saved game
- Multiple Saved Games
    - Generic image representing multiple saved games
    - Total size of all of the currently selected saved games
- Game Title
    - 2D image associated with the game
    - Name of the game
    - Total number of saved games
    - Total size in blocks of the game (sum of saved games, persistent data, etc.)

In one implementation, game developers provide certain information relating to each saved game such that the user of the gaming system can easily identify the

saved game they want to play or copy onto another storage device, such as memory unit 140. An example of the information that a particular application may save when a save game command is executed might include:

- Name of the game associated with the saved game data
- A graphic representation of the location in the game when saved
- A brief description of the game state when saved (e.g., Level 3- At the Castle Entrance)
- Date and time the game was saved

In other implementations, more or less data may be saved in response to a save game command.

In a particular implementation, the gaming system requires the game developer to use a descriptive name when saving a game. A descriptive name is something other than, for example, "save.001" or "savegame.b". Instead, a descriptive name requires, for example, an identification of the game that created the saved game file and some type of information relating to the status of the game and/or the date and time the game was saved. To accommodate this descriptive name, game developers are permitted to create saved game names of any character length and saved game names can use any characters (including symbols, punctuation, etc.). Providing descriptive names with saved games allows the user of the gaming system to easily locate the desired saved game.

A top title pane 1010 provides summary information, such as a friendly name of the storage device (e.g., "Steve's Games"), the memory device's total

storage space in blocks, and the memory device's storage space left in blocks. Select and back elements support navigation to other screens.

Although the invention has been described in language specific to structural features and/or methodological acts, it is to be understood that the invention defined in the appended claims is not necessarily limited to the specific features or acts described. Rather, the specific features and acts are disclosed as exemplary forms of implementing the claimed invention.